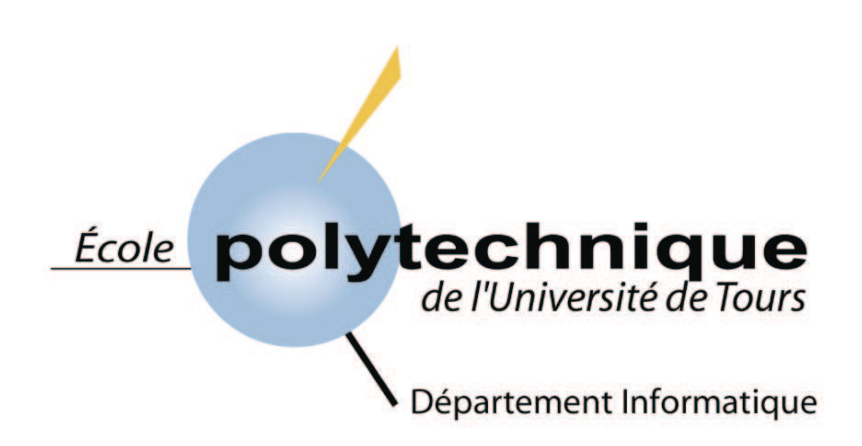
UNIVERSITE FRANÇOIS RABELAIS TOURS
POLYTECH'TOURS - DÉPARTEMENT INFORMATIQUE
64, Avenue Jean Portalis
37200 TOURS



Projet de Robotique

Asservissement de la position de l'organe terminal d'un robot sur un capteur 3D.

E.P.U.-D.I.2

Janvier 2006

Encadré par :
M. Pascal MAKRIS
M. Arnaud PURET

Étudiants :
AGEN Florian
MICHOT Julien
DI2 Promotion 2004 - 2007

Asservissement de la position de l'organe terminal d'un robot sur un capteur 3D

Florian AGEN - Julien MICHOT

Janvier 2006

Introduction

L'asservissement de la position de l'organe terminal d'un robot sur un capteur trois dimensions est la première étape d'un projet beaucoup plus complexe d'aide à la chirurgie dans le domaine médical.

A partir d'un robot présentant plusieurs axes de liberté, il s'agit de pouvoir amener les instruments médicaux présentés par le robot à l'endroit exact voulu par le chirurgien. Plusieurs possibilités restent envisagées quant à la façon de déterminer cette position. Nous avons ici choisi de la modéliser à l'aide d'un gant équipé de plusieurs capteurs.

Cette présente étude, menée dans le cadre du projet de robotique de deuxième année du Département Informatique de l'École Polytechnique de l'Université de Tours, doit pouvoir servir de vitrine sur nos capacités et nos possibilités en la matière.

Dans cette optique, toute une modélisation de type Génie Logiciel a été mise en place. A partir du cahier des charges, nous détaillerons les étapes de conception et présenterons le logiciel qui en résulte à travers son guide de développement et son guide d'utilisation.

Table des matières

Chapitre 1

Définition du problème

1.1 Le cahier des charges

Le cahier des charges s'articule autour de trois volontés principales :

1. Concevoir une interface à la fois belle et conviviale, qui puisse attirer l'oeil et intéresser les personnes étrangères au monde de l'informatique. Le programme doit pouvoir être utilisé lors des manifestations où l'école, et le Laboratoire d'Informatique sont représentés.
2. Fournir un suivi de précision, qui démontre nos capacités dans le domaine de la robotique. Le projet ne peut prétendre être le point de départ à l'étude complète d'aide à la chirurgie. Toutefois, la conception doit être suffisamment précise et documentée pour pouvoir aider à la mise en place rapide de nouveaux procédés.
3. Assurer l'aspect pédagogique nécessaire à la formation d'ingénieurs, à travers la découverte de nouveaux périphériques et de méthodes de conception.

Aucune restriction sur le matériel n'est précisée. Le robot *SCORBOT ER V Plus*, de la société **ESHED ROBOTEC** sera commandé sur la position du capteur 3D Glove P5 prêté pour l'étude par l'équipe RTIC du Laboratoire d'Informatique. Ces deux périphériques seront détaillés dans la prochaine partie.

1.2 Le matériel utilisé

1.2.1 Le robot SCORBOT ER V Plus

Le *SCORBOT ER V Plus* est un bras manipulateur robotisé comportant 5 axes de liberté. Utilisé initialement en chaîne de production, il est capable de mémoriser des positions pour les répéter par la suite, ou de suivre les ordres qui lui sont transmis. C'est dans cette dernière configuration que nous devons l'utiliser.

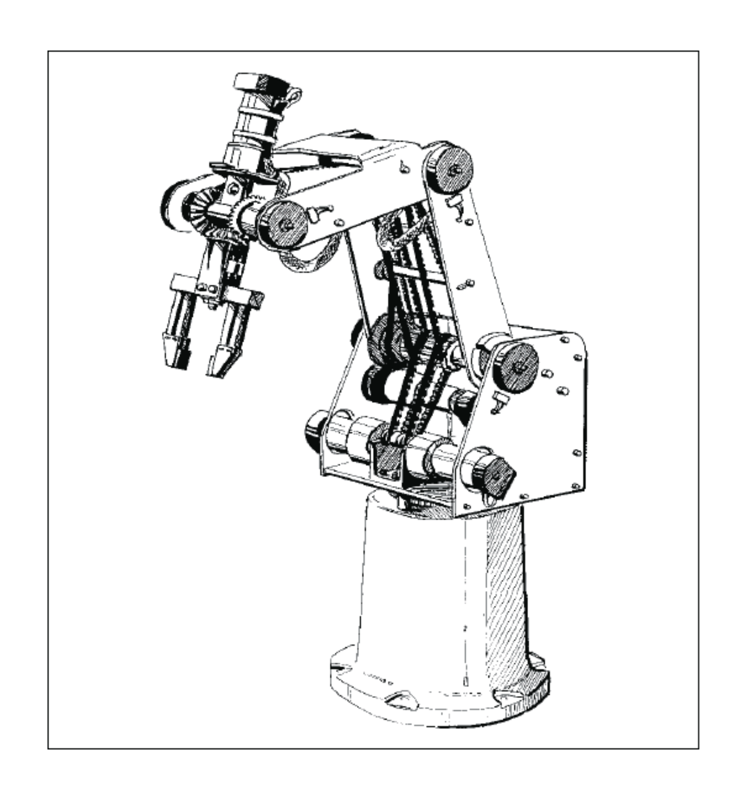


FIG. 1.1 – Le SCORBOT ER V Plus

Le robot fonctionne à l'aide de moteurs commandés. Des capteurs de sécurité donnent les valeurs maximales de buté pour empêcher aux axes de se détériorer, et des codeurs fixés sur l'origine de chaque axe fournissent les informations sur la position des éléments par rapport à la base.

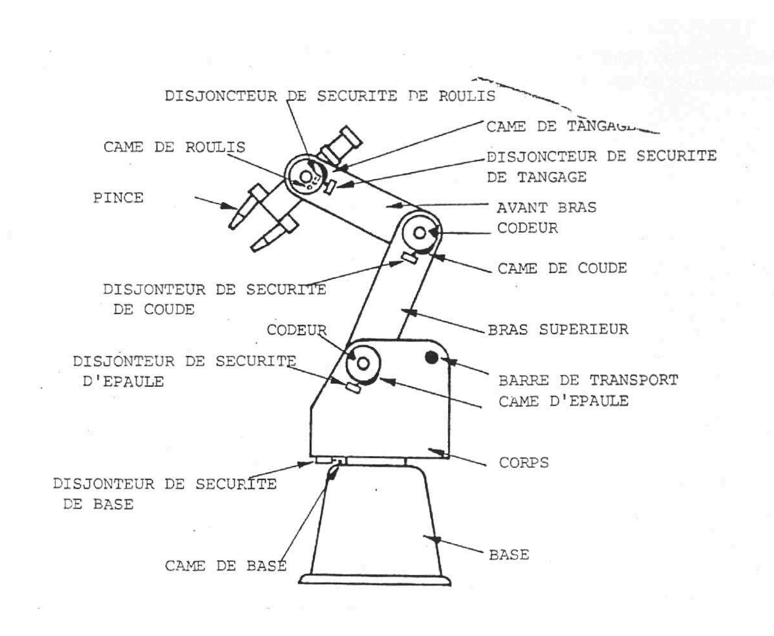


FIG. 1.2 – Description des éléments du SCORBOT

Cette technologie présente cependant l'inconvénient d'être fragile et facilement perturbée. Le fait de changer ne serait-ce qu'un peu la position des codeurs en manipulant le robot change également les valeurs des axes du repère d'origine, et la position des butés.

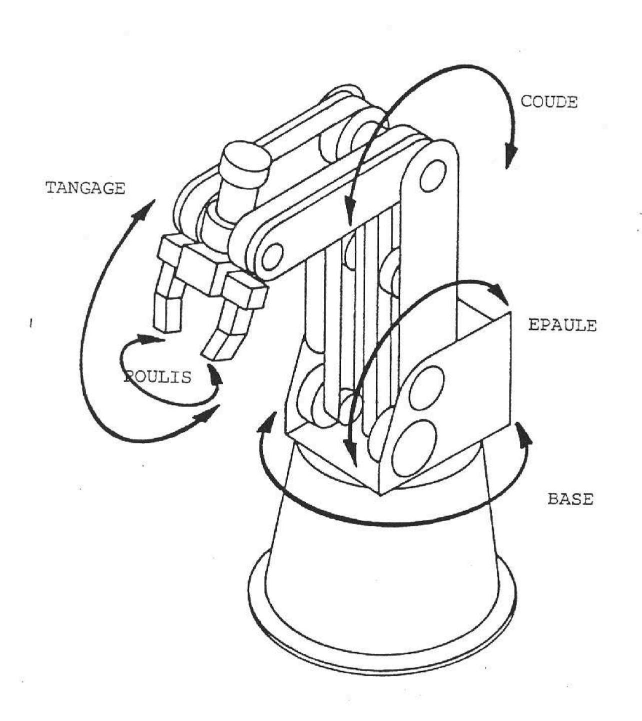


FIG. 1.3 – Mouvements des différents axes

Le robot que nous utilisons pour ce projet a une vingtaine d'années, et est donc sensiblement moins performant que lors de leur mise en service. Les courroies qui entraînent le bras lors de mouvements sont usées, et nous sommes obligés de fixer des valeurs d'amplitude pour le déplacement inférieure à ce que le permet le guide de fonctionnement du SCORBOT.

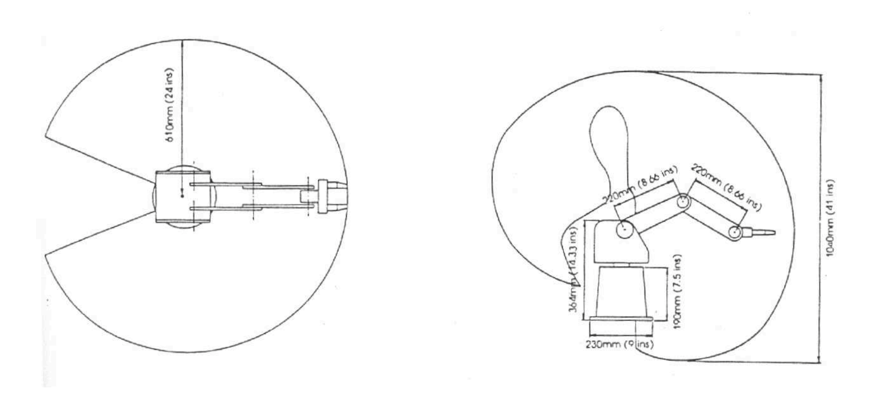


FIG. 1.4 – Amplitude possible des mouvements

Il est possible de commander le robot à travers ses coordonnées dans un repère cartésien, en X Y Z pour la position de l'organe terminal, ainsi que le roulis et le tangage de la pince comme le ferait une main.

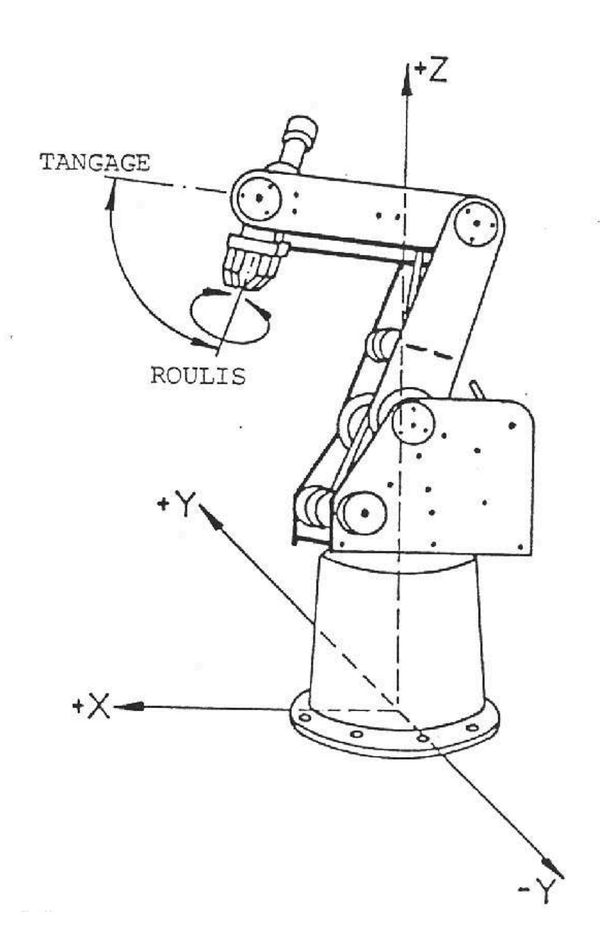


FIG. 1.5 – Position du repère de déplacement

Le SCORBOT communique avec l'ordinateur par le port Série RS232, utilisable directement en le configurant comme le spécifie la documentation constructeur : mode XON/XOFF, vitesse de 9600 bauds, 8 bits de données, 1 bit d'arrêt, et pas de parité. Pour la programmation, il faut passer par le langage ACL, Langage Avancé de Contrôle, imposé par **ESHED ROBOTEC** pour l'interaction avec le robot. Cette partie de l'étude est détaillée dans le guide de développement, ainsi que dans le manuel de référence fournit avec ce rapport.

1.2.2 Le capteur 3D

Le gant 3D P5 est un périphérique USB conçu au départ pour augmenter la sensation des joueurs de jeux vidéo lors de contrôles nécessitant une vrai interaction avec l'univers 3D. Basé sur des capteurs de position présents à la fois dans la main et dans les doigts, le P5 se sert d'une borne infrarouge pour positionner son référentiel dans l'espace.



FIG. 1.6 – Le Glove P5

Ce périphérique disponible uniquement aux États-unis depuis quelques mois, reste un capteur 3D bon marché. Son faible coût traduit la simplicité et la fiabilité de la technologie employée. Le gant peut servir de contrôleur de pointeur de souris sous les systèmes d'exploitation Windows, mais également de référentiel 3D comme nous l'utilisons avec le robot.

Le périphérique est destiné principalement pour Windows. Le CD fournit originellement ne contient que les pilotes pour ce système d'exploitation, et le constructeur ne mentionne pas son comportement sous d'autres systèmes, tel UNIX. Le kit de développement téléchargeable via internet donne une description sommaire des méthodes mises à la disposition des développeurs, mais reste suffisante pour prendre en main le P5 facilement, et l'intégrer à nos outils de développement.

La fiabilité du système reste le point noir de ce capteur 3D. Nous le verrons particulièrement lors de son utilisation pour asservir le robot SCORBOT dans la dernière partie de ce rapport. Pour le contrôle de la souris déjà, la stabilité du pointeur reste approximative. Le pointeur présente des sauts de position souvent inattendus, qui sont expliqués dans le manuel d'utilisation du contrôleur par une lumière externe intense ou une distance trop importante vis à vis du récepteur. Le dispositif est donc très sensible à l'environnement, et l'on peut déjà entrevoir les problèmes que vont nous poser ces désagréments lors de l'asservissement du robot.



FIG. 1.7 – Le gant 3D

1.3 Spécifications du système

Les deux périphériques utilisés peuvent fonctionner sous les systèmes d'exploitation *Windows* et *Linux*. Cependant, le gant 3D *P5* est fourni avec un pack de développement téléchargeable via le site internet du constructeur, conçu uniquement pour *Windows*. Nous utilisons donc ce système d'exploitation pour le développement du projet.

Un langage objet semble particulièrement indiqué pour l'étude, puisque une programmation modulaire permet par la suite de ne remplacer qu'une partie du programme, en gardant par exemple l'interface et la gestion de la communication. Le langage C++ est donc utilisé pour le développement. De plus, nous développons sous *Microsoft Visual Studio .NET 2003*, pour donner un aspect nouveau à l'utilisation des robots *SCORBOT*, naturellement programmés pour les systèmes d'exploitation DOS sous *TURBO C++*. Pour ne pas perdre en portabilité, l'interface est réalisée à l'aide de la librairie graphique *QT 3.3.3* de *Trolltech*, et facilite donc son adaptation sous les systèmes *UNIX*.

A la suite des premiers tests d'utilisation des robots *MINI S* proposés premièrement, et des *SCORBOT* par la suite, nous nous sommes aperçu que les communications séries pausaient souvent des problèmes de fiabilité. Il faut donc pouvoir tester la communication avec le robot indépendamment de l'asservissement sur le capteur 3D, afin d'être sûr que l'organe terminal puisse se déplacer selon notre souhait. L'interface comprend pour cela une ' télécommande ' de test des cinq axes du robot placée dans le coin supérieur droit de l'interface.

D'autre part, le cahier des charges nous impose une certaine convivialité dans la façon dont l'asservis-

sement est perçu par l'utilisateur. Nous avons choisi pour cela de placer une fenêtre de visualisation 3D représentant à la fois la position du capteur, et celle de l'organe terminal, réalisé en *OpenGL*, toujours par souci de portabilité. Toutes les informations concernant l'asservissement sont donc disponibles sur l'interface. Cette dernière respecte la majorité des règles d'IHM, et propose ainsi un contrôle attractif du manipulateur, comme le montre le guide d'utilisation présenté en dernière partie de ce rapport.

Chapitre 2

Guide de développement

Ce guide de développement a pour but de faciliter la reprise du programme *AssertRobot* lors de modifications ou d'ajouts d'améliorations futures. Plusieurs parties sont détaillées. Nous présentons tout d'abord la gestion de la communication avec le robot, et quelques exemples de commande du manipulateur, puis l'interface du programme et la gestion du gant 3D. Le manuel de références donné en annexe de ce rapport contient l'ensemble des modélisations UML, ainsi que les diagrammes de classe et explications sur l'ensemble des membres et des méthodes qui les composent. Un développeur ayant déjà une expérience de la programmation QT n'aura donc aucun mal à ajouter des fonctionnalités à ce projet.

2.1 Gestion de la communication série RS232

La configuration du port série RS232 pour la communication avec le robot n'est en rien différente de ce qui est étudié pendant les cours de périphérique spécialisé de deuxième année. Nous utilisons deux *handles* pour ouvrir la communication en mode lecture et écriture générique, conformément à ce que suggère le guide d'utilisation du robot SCORBOT. Nous rappelons également ici les options de configuration : mode XON/XOFF, vitesse de 9600 bauds, 8 bits de données, 1 bit d'arrêt, et pas de parité. Le détail sur la méthode est donné dans le manuel de références, notamment page 62 et 65 où la méthode *serie_config* de la classe *RobotScorbot* est détaillée. L'ensemble du code et des classes se rapportant au robot et à la communication série est regroupé dans les fichiers *RobotScorbot.h* et *RobotScorbot.cpp*.

L'envoi des données au robot se fait par l'intermédiaire d'une chaîne de caractère écrite directement sur le port série. Le message envoyé correspond aux critères syntaxiques imposés par le langage ACL. Il se compose en général d'un mot spécifique suivi de plusieurs paramètres. Il est a noter que le robot renvoi le caractère '>' après exécution de la commande demandée, et non après réception du message.

Nous définissons une position comme un enregistrement composé de 6 champs : 3 pour définir les coordonnées cartésiennes X Y Z, 2 pour donner une valeur de roulis R et de tangage T, et une chaîne de caractères permettant de définir le nom de chaque position. Cette structure nous est encore une fois imposée par le Langage de Contrôle Avancé.

Le SCORBOT est capable de comprendre une centaine de commandes enregistrées dans son unité de contrôle, mais nous n'en n'utilisons que certaines :

– le HOMING, défini par le message *"home\r"* qui lance le programme interne de test des axes du robot. Ce programme permet au SCORBOT de se réinitialiser, mais demande un certain temps.

– La demande de position des axes, définie par le message *"listpv NomPosition \r"* entraînant l'envoi par le SCORBOT d'une chaîne de caractères contenant tes les informations de position courante, tangage et roulis de son organe terminal.

– Les mots clés *defp* et *teach*, qui enregistrent les nouveaux paramètres X Y Z R et T, et envoient au robot une demande de déplacement vers cette nouvelle position.

– *open* et *close* (respectivement *"open\r"* et *"close\r"*) pour ouvrir et fermer la pince du robot.

Ainsi, il suffit de passer une chaîne de caractères correspondante à une commande ACL en paramètre de la fonction *serie_envoyer* pour que le robot la prenne en compte. Pour le *homing* par exemple, on écrira *serie_envoyer("home\r")*. Le mode de communication développé pour l'application permet évidemment d'envoyer ou de recevoir l'ensemble des commandes disponible dans le langage ACL.

2.2 Programmation du glove P5

Le glove P5 peut être programmé en différents langages, mais le kit anglais *P5 SDK 2.0* ne contient que les méthodes pour les langages C et C++. Le pack présente également 7 exemples de programmation dont le code est fourni, ainsi qu'un fichier d'aide au format *chm* pour prendre en main l'API du P5.

On peut distinguer deux grandes catégories de méthodes : le *P5 Bend*, qui permet d'implémenter les fonctions pour la prise en compte des doigts du gant, et le *P5 Motion*, qui donne la possibilité aux programmeurs de récupérer la position de la main, ainsi que les paramètres de déviation, tangage et roulis (*yaw, pitch and roll*).

Dans tous les cas, l'API utilise la librairie dynamique *P5DLL.dll*, qui doit être disponible sur l'ordinateur pour que l'application puisse être compilée. Pour pouvoir utiliser les paramètres des doigts, certaines étapes doivent être respectées :

1. Inclure les fichiers *P5Bend.h* et *P5Bend.cpp* dans le projet en cours.
2. Comme pour tout fichier d'en-tête, ajouter *#include "P5Bend.h"* dans le fichier où les données du P5 seront utilisées.
3. Dans la fonction d'initialisation, comme dans le constructeur de la classe où est pris en compte le P5 par exemple, ajouter les lignes *P5Bend_Init(&P5, 0) ; P5Bend_SetClickSensitivity(P5_THUMB, 10) ;* où P5 doit être un pointeur sur une variable de type CPDLL, et 10 est la valeur de la constante de sensibilité pour le pouce (*THUMB*).
4. Dans la boucle principale de l'application, appeler la fonction *P5Bend_Process* pour mettre à jour la valeur des différentes données relatives aux doigts.

De la même façon, l'utilisation de *P5Motion* est soumise à quelques étapes :

1. Inclure les fichiers *P5Motion.h* et *P5Motion.cpp* dans le projet en cours.
2. Ajouter *#include "P5Bend.h"* dans le fichier où les données du P5 seront utilisées.
3. Dans la fonction d'initialisation, ajouter la ligne *P5Motion_Init(&P5, 0) ;* où P5 doit être un pointeur sur une variable de type CPDLL.

4. Dans la boucle principale de l'application, appeler la fonction *P5Bend_Process* pour mettre à jour la valeur des différentes données relatives au gant.

Toutes ces instructions sont détaillées dans l'implémentation de la classe *GLObjectWindow* dont la définition se trouve dans le fichier *AssertRobotWin.cpp*. D'autres fonctions de configuration sont disponibles. Elles sont détaillées dans le fichier d'aide du kit de développement.

Il est également possible d'utiliser trois des quatre boutons présents sur le gant pour effectuer des actions prédéfinies. Il suffit d'inclure dans la condition le code *P5->m_P5Devices[0].m_byButtons[#]* où # est le numéro du bouton poussoir.

2.3 Présentation de l'interface

L'interface développée grâce au GUI QT s'articule autour d'une fenêtre principale sur laquelle les différents contrôles et éléments de visualisation sont ajoutés. Les fichiers *AssertRobotwin.h* et *AssertRobotwin.cpp* contiennent l'ensemble des déclarations et définitions des éléments décrits sur la figure si-dessous :

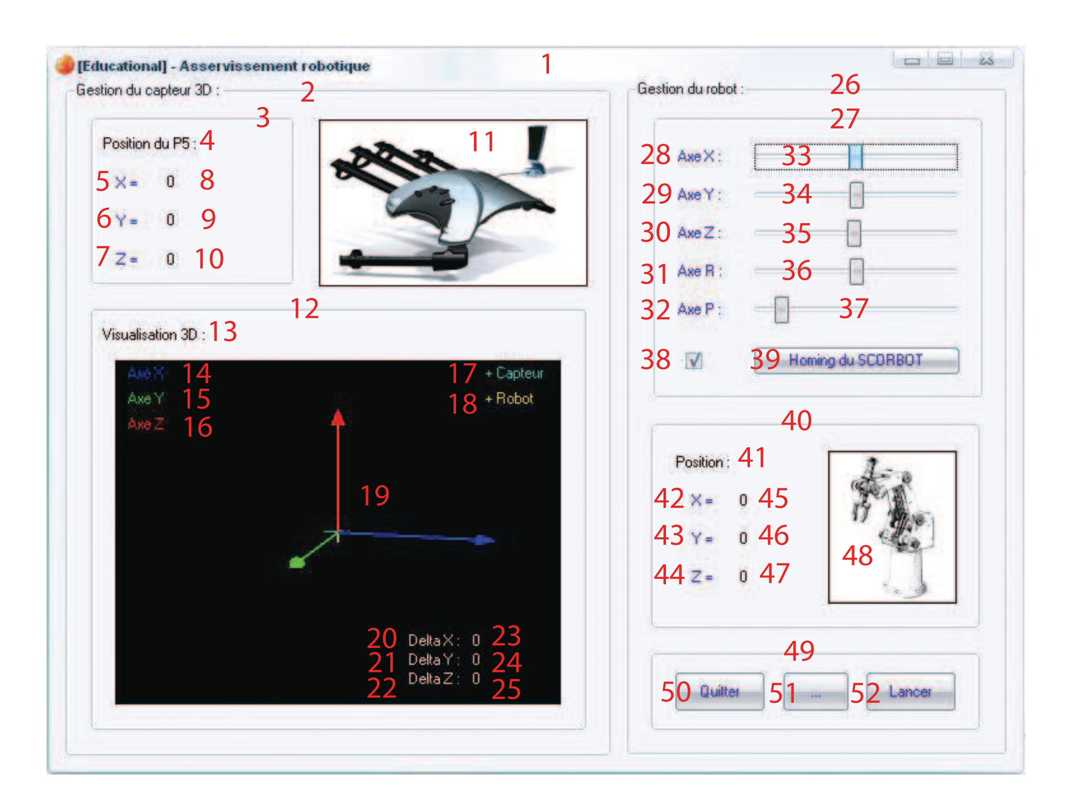


FIG. 2.1 – L'interface du programme

Numéro	Nom de l'élément	Type	Numéro	Nom de l'élément	Type
1	GLObjectWindow	*QWidget*	27	CtrlRobotFrame	*QGroupBox*
2	CapteurBox	*QGroupBox*	28	AxeXtextLabel	*QLabel*
3	CapteurFrame	*QGroupBox*	29	AxeYtextLabel	*QLabel*
4	LabelP5	*QLabel*	30	AxeZtextLabel	*QLabel*
5	XtextLabel	*QLabel*	31	AxeRtextLabel	*QLabel*
6	YtextLabel	*QLabel*	32	AxePtextLabel	*QLabel*
7	ZtextLabel	*QLabel*	33	AxeXSlider	*QSlider*
8	XLabelP	*QLabel*	34	AxeYSlider	*QSlider*
9	YLabelP	*QLabel*	35	AxeZSlider	*QSlider*
10	ZLabelP	*QLabel*	36	AxeRSlider	*QSlider*
11	PhotoP5Glove	*QLabel*	37	AxePSlider	*QSlider*
12	OpenGlFrame	*QGroupBox*	38	CmdRobotCheckBox	*QCheckBox*
13	V3DtextLabel	*QLabel*	39	InitRobotButton	*QPushButton*
14	CouleurXLabel	*QLabel*	40	PositionRobotFrame	*QGroupBox*
15	CouleurYLabel	*QLabel*	41	textLabelPositionR	*QLabel*
16	CouleurZLabel	*QLabel*	42	XtextLabelR	*QLabel*
17	CaptextLabel	*QLabel*	43	YtextLabelR	*QLabel*
18	RobtextLabel	*QLabel*	44	ZtextLabelR	*QLabel*
19	c	*GLBox*	45	XLabelR	*QLabel*
20	DeltaXText	*QLabel*	46	YLabelR	*QLabel*
21	DeltaYText	*QLabel*	47	ZLabelR	*QLabel*
22	DeltaZText	*QLabel*	48	PhotoScorbot	*QLabel*
23	DeltaXLabel	*QLabel*	49	ControleFrame	*QGroupBox*
24	DeltaYLabel	*QLabel*	50	QuitterButton	*QPushButton*
25	DeltaZLabel	*QLabel*	51	AboutButton	*QPushButton*
26	CtrlRobotBox	*QGroupBox*	52	LancerButton	*QPushButton*

2.4 Phases de codage

Une fois l'interface créée, la communication avec le robot établie, et les fonctions du gant implémentées, plusieurs problèmes restent à résoudre. Les butés mécaniques du robot ne sont pas toutes fonctionnelles, et il est nécessaire d'établir des valeurs minimales et maximales pour chaque axe du bras. Il est également important de définir une position de départ pour l'organe terminal du robot, dans le cas fréquent où une initialisation deviendrait indispensable suite à un changement brutale de la position indiquée par le P5.

Les domaines de définition des axes du SCORBOT et du gant ne sont pas les mêmes. Pour que l'asservissement se fasse entre les deux périphériques, il faut donc appliquer une normalisation qui doit également apparaître sur la fenêtre de visualisation 3D, sans quoi il serait impossible de pouvoir commenter le fonctionnement.

L'affichage des informations sur l'interface est rafraîchie au moyen d'un timer, déclenché toutes les 50 ms, qui interroge à la fois le capteur 3D et le robot sur leur position respective. Le SCORBOT ne peut renvoyer cette information que s'il n'est pas déjà entrain de prendre en compte une autre instruction, ce qui, le cas échéant, fige le système pendant une durée indéterminée, et surtout non souhaitée. Deux *threads* ont donc été implémentés pour palier le problème. Le premier gère la communication avec le robot pendant que le deuxième se consacre à la mise à jour de l'affichage. Les figeages sont donc évités.

La conception respecte dans la mesure du possible, les principes du Génie Logiciel. Chaque classe

possède son propre fichier de déclaration et de définition. Ainsi, *AssertRobot* se compose :

- Des fichiers *AffichageThread.h* et *AffichageThread.cpp* pour la gestion des méthodes du thread d'affichage,
- *AssertRobot.h* et *AssertRobot.cpp* pour la fenêtre OpenGL,
- *AssertRobotWin.h* et *AssertRobotWin.cpp* pour l'interface QT,
- *CommunicationThread.h* et *CommunicationThread.cpp* pour la gestion du thread de communication avec le robot,
- *Main.cpp*
- *P5Bend.h*, *P5Bend.cpp*, *P5Motion.h*, *P5Motion.cpp*, *P5dll.h* et *P5dll.lib* pour l'interaction avec le gant 3D P5,
- *RobotScorbot.h* et *RobotScorbot.cpp* pour l'interaction avec le robot SCORBOT.

2.5 Manuel de références

Le manuel de référence, fournit en annexe de ce rapport à la fois en format numérique et papier, contient toute la modélisation UML et la description des différentes classes illustrée par le code source. Ce document a pour but de compléter ce guide de développement, et ainsi d'accélérer l'implémentation de composants complémentaires au logiciel *AssertRobot*.

Chapitre 3

Guide d'utilisation

Nous ne nous plaçons plus ici du point de vue du concepteur, mais de l'utilisateur du système pour décrire son fonctionnement.

3.1 Présentation du programme

Le programme comprend deux modes de fonctionnement. Le mode ”télécommande”, qui permet à l'utilisateur de tester la communication avec le robot en jouant sur les différents axes, et le mode ”asservissement” pour lequel l'organe terminal du robot va répéter les mouvements indiqués par la main.

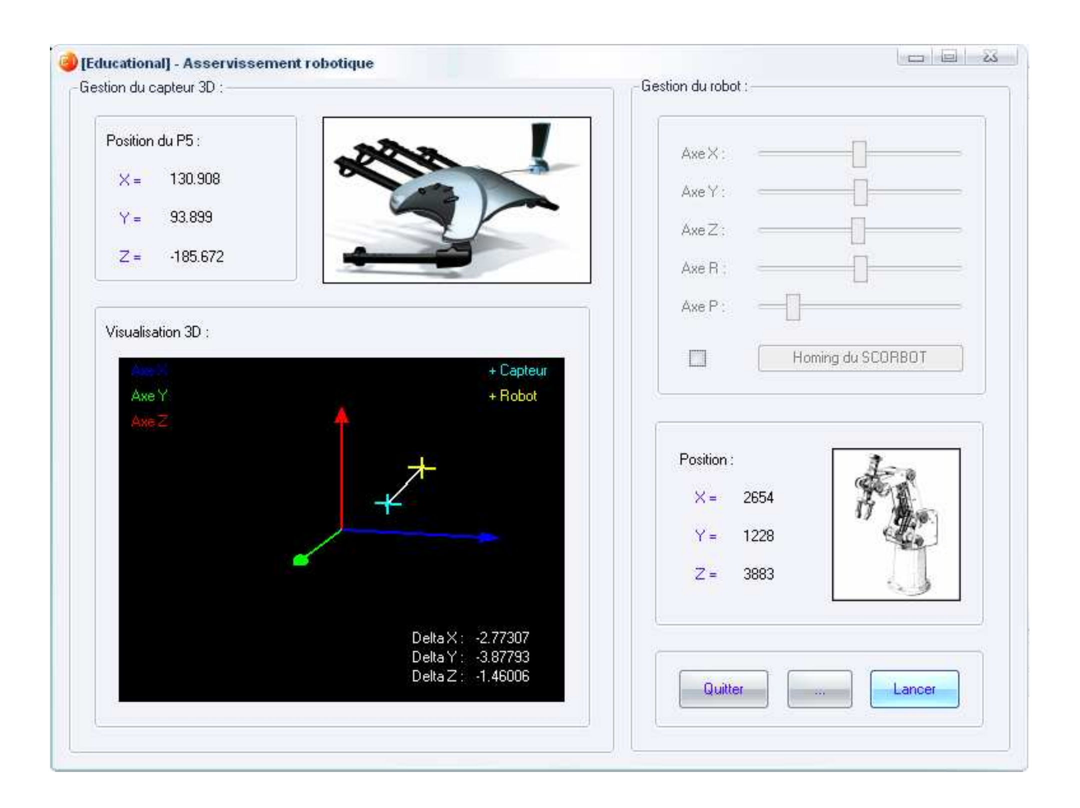


FIG. 3.1 – Le logicile *AssertRobot* en fonctionnement

Plusieurs informations apparaissent sur l'interface :

– La position courante du P5 en coordonnées réelles est indiquée en haut à gauche de la fenêtre, et une croix de couleur bleu turquoise permet de visualiser dans la représentation 3D son emplacement par rapport à l'organe terminal du robot.

– La position de l'organe terminal est affichée à côté de la photo du SCORBOT dans son propre système de coordonnées, et la croix jaune indique son déplacement lors de l'asservissement.

– La distance normalisée entre les deux entités est représentée par le trait blanc entre les deux croix, ainsi que par les *deltaX*, *deltaY* et *deltaZ*, donnant les valeurs exactes du retard de la pince du SCORBOT sur le P5.

Le bouton ”Lancer” active le mode”asservissement”. Il faut donc s'assurer d'être prêt à contrôler le robot avant de l'activer, car le manipulateur suivra immédiatement les mouvements indiqués par le gant. Pour revenir en mode ”télécommande”, il suffit de cocher la case prévue à cet effet située à côté du contrôle ”Homing du SCORBOT”. Ce dernier, comme son nom l'indique, active le programme d'initialisation du robot avant de le remettre dans la position indiquée par la télécommande.

3.2 Mise en condition avant utilisation

Le bon fonctionnement du système dépend grandement de l'installation préalable des deux périphériques. Pour que le logiciel puisse démarrer, il faut évidemment que le gant et le robot soient connectés à l'ordinateur. Le SCORBOT ne demande pas plus qu'un simple branchement de son connecteur sur le port COM1, mais le P5 doit tout d'abord être installé, et configuré pour pouvoir être utilisé. Il est notamment nécessaire d'étalonner la position des doigts pour que, lors de l'asservissement, la pince ne se ferme pas inopinément. De plus, la position de la borne réceptrice du capteur 3D ne doit pas être négligée. Elle doit être suffisamment près des mouvements pour les détecter, sans toutefois réduire le champ de fonctionnement du capteur pour la prise en compte des mouvements. Il est a rappeler que, comme nous l'avons déjà signalé lors de la présentation de ce périphérique, une faible luminosité reste un facteur de précision.



FIG. 3.2 – Mise en place de la borne 3D

3.3 Comprendre le code des erreurs

Au cours de l'utilisation du programme *AssertRobot*, ou lors de son initialisation, certaines erreurs peuvent apparaître. La plupart sont dues à une mauvaise mise en condition avant utilisation, et ce code des erreurs doit pouvoir permettre de remédier à la majorité des oublis :

Code de l'erreur	Message	Cause	Résolution
E0001	Aucun périphérique P5 détecté	L'ordinateur n'arrive pas a accéder au périphérique P5	Vérifier que le capteur 3D est bien connecté a un port USB de l'ordinateur, et que la lumière ROUGE du récepteur est bien allumée, si le problème persiste, se référer au guide d'utilisation du P5
E0002	Erreur d'ouverture du port série	Le port COM1 est indisponible	Cela arrive parfois quand le robot n'arrive plus à gérer la communication. Arrêter le programme et relancer le. Si besoin redémarrer l'ordinateur.
E0003	Erreur de configuration du port série	Le port COM1 ne peut pas être configuré	Vérifier qu'aucun autre programme n'est en-train d'utiliser le port série
E0004	Impossible de lancer le timer	Le timer qui gère l'asservissement n'a pas pu être lancé	Relancer le programme

Le fait d'utiliser sous *Windows XP* le robot SCORBOT prévu initialement pour les systèmes d'exploitation *DOS* et les premiers *Windows* pause quelques problèmes d'adaptation. Le port COM se bloque de temps en temps et l'initialisation de la communication devient impossible. Cette erreur est cependant rare et dépend également du nombre de programmes utilisant le port de communication pouvant interférer dans l'ordinateur.

Chapitre 4

Remarques et Résultats

Cette dernière partie présente les résultats obtenus lors de l'asservissement de la pince du robot sur le capteur 3D P5, ainsi que les possibilités d'évolution du projet, d'un point de vue technique et technologique.

4.1 L'asservissement réalisé

Le but initial de cette étude était de pouvoir réaliser une recopie de mouvements indiqués par la main de l'Homme sur un manipulateur robotique présentant les mêmes degrés de liberté. Le logiciel *AssertRobot,* conçu pour ce projet, répond parfaitement à cette attente. En plus de permettre de suivre le mouvement, l'utilisateur est capable de saisir un objet sur un plan de travail, et ce après quelques minutes d'accommodation au manipulateur. Un mouvement lent et adapté à la réponse du robot est néanmoins nécessaire pour garantir la saisie.

Les outils mis à notre disposition ont pourtant demandé certains choix quant à l'implémentation. Le manque de précision, et le changement parfois soudain de la position du capteur altèrent la stabilité de l'organe terminal lors de la saisie des objets. Un seuil permet de réduire les "tremblements" de la pince, mais diminue la fluidité de l'asservissement.

La communication avec le SCORBOT n'est pas non plus adaptée à un système de contrôle "en temps réel", comme le voudrait la théorie. Le langage ACL, par lequel il faut passer pour envoyer des ordres au robot, simplifie son fonctionnement, mais doit aussi ralentir son exécution. Chaque nouvelle position envoyée au manipulateur est dans un premier temps enregistrée dans l'unité de contrôle, avant d'être exécutée par le SCORBOT. Celui-ci est donc très précis dans son positionnement, mais met du temps à prendre en compte les nouvelles coordonnées. Le fait de ne posséder qu'un seul canal de communication pour à la fois demander au robot sa position, et lui indiquer la marche à suivre quand le robot n'est fait que pour faire une seule chose à la fois ralenti peut-être aussi le système.

4.2 Améliorations possibles

Le projet est parfaitement fonctionnel, mais certaines améliorations peuvent encore être apportées. Nous les déclinons en deux parties : les améliorations matérielles, que nous décrivons dans une dernière section, et les perfectionnements sur le logiciel *AssertRobot*, que nous n'avons volontairement pas mis en

place, afin de garder le temps nécessaire à la rédaction d'une documentation de qualité.

Tous les axes de liberté du robot n'ont pas été exploités, notamment le roulis et le tangage de la pince, bien que le gant P5 permette pourtant de jouer sur ces deux paramètres. En terme de programmation, leur implémentation est identique à celle réalisée pour les coordonnées X, Y et Z, et ne présente donc aucune difficulté. Il faut simplement prendre le temps de fixer les valeurs maximales des deux nouveaux paramètres qui apparaissent déjà dans le programme.

Le gant P5 demande un étalonnage avant utilisation, que nous faisions par l'intermédiaire d'un logiciel annexe installé en même temps que les pilotes du périphérique. Nous aurions pu implémenter une fenêtre de paramétrage du gant, qui aurait rendu son utilisation autonome de tout autre application.

Pour finir, un pack de développement du P5 est également disponible depuis peu pour les systèmes d'exploitation UNIX, il aurait donc été intéressant de faire une version LINUX du logiciel d'asservissement pour voir comment ce périphérique se comporte dans un autre environnement.

4.3 Adaptabilité du projet

L'avenir du projet d'un point de vu scientifique passera indéniablement par l'utilisation de périphériques à la fois plus performants et plus précis.

Le capteur 3D P5 est bon marché. Il était initialement prévu comme substitut de la souris pour les jeux vidéo et la bureautique, et ne convient pas tout a fait au contrôle d'entités demandant une grande fiabilité pour la sécurité des utilisateurs. Le bras manipulateur peut en effet se révéler dangereux s'il effectue des mouvements brusques dans un périmètre de fonctionnement non sécurisé, et nous ne pouvons nous permettre d'utiliser des capteurs trop sensibles à la luminosité. D'autres systèmes de ce type existent sur le marché, et seraient sans doute beaucoup plus appropriés.

Le robot à l'avantage d'être très précis dans son positionnement, mais la communication série est peut-être un peu désuète aujourd'hui, quand il faudrait un manipulateur pouvant interpréter les ordres quasi-instantanément. L'application de cette étude à la chirurgie demandera la conception d'un nouveau robot répondant à cette attente, et le SCORBOT peut convenir, en attendant, de périphérique de test pour le positionnement.

Conclusion

L’asservissement de la position de l’organe terminal du robot *SCORBOT ER V* sur le capteur 3D *P5* est un succès compte tenu des limitations offertes par le système. L’utilisateur est désormais capable, avec un peu d’entraînement et des conditions de luminosité favorables à l’utilisation du gant, de prendre un objet sur le plan de travail, et de le déplacer dans l’espace. Le manipulateur suit parfaitement le mouvement commandé par la main à la condition que celui si ne soit pas trop brusque, et que l’utilisateur s’adapte aux temps de latence imputé à la liaison série.

Le logiciel développé pour l’étude est donc parfaitement fonctionnel. Sa fenêtre de visualisation OpenGL apporte un aspect ludique et attracteur au projet, et rend encore plus convivial son utilisation. Les deux périphériques utilisés restent malheureusement bien trop approximatifs pour espérer pouvoir les intégrer dans un but plus scientifique. Le P5 glove est trop imprécis pour contrôler de façon optimal le robot, et le SCORBOT, qui date tout de même d’une vingtaine d’années, présente tous les inconvénients des technologies aujourd’hui en décalage avec ce qui se fait à l’heure actuelle. Il est donc nécessaire d’investir dans un nouveau dispositif de localisation 3D pour obtenir un asservissement de précision.

Table des figures

Bibliographie

– SCORBOT ER V, SCORBOT ER V Plus Manuel d'utilisateur 1ère ed. 100027 - 1982 - ESHED ROBOTEC

– Guide de référence ACL : Langage Avancé de control 1ère ed. 100027 - 1982 - ESHED ROBOTEC

– Site web du P5 Glove *http ://www.alliancedistributors.com/Alliance_Brand/Products/Developers.php*

– Rapport projet de SE : Le bras robot - Juin 2005 - L.Moisant, J.Teindas

Résumé

Le contrôle de l'organe terminal d'un manipulateur robotique par un capteur 3D est le prémisse d'une étude sur la conception de robots pour la chirurgie. Le contrôleur P5 glove, un nouveau périphérique basé sur la réception dans l'espace de signaux infrarouges envoyés par des capteurs de position, nous permet de diriger le robot SCORBOT ER V Plus utilisé pour l'étude. Il est important de pouvoir tester la fiabilité de tels systèmes avant de poursuivre sur la conception de robots personnalisés, et le logiciel AssertRobot, spécialement développé dan ce but, montre nos possibilités en la matière. L'utilisateur est capable par l'intermédiaire du gant de diriger le robot pour la préhension d'objets sur un plan de travail, ou encore de reproduire des mouvements de précision.

Mots clés

Robotique, SCORBOT, Capteur 3D, P5 glove, asservissement.

Abstract

The robot system terminal control by a 3D sensor is the first part of a most important project about the aided surgery thanks to robotic. The P5 Glove controller, a brand new 3D peripherical, points out the position to a hand-like manipulator : the SCORBOT ER V Plus of ESHED ROBOTEC. That's important to test the reliability of 3D sensors before to make choices and continue with surgery purposes, and the developped software allows to check our possibilities in the matter. We developped an OpenGL window to visualize the position of the two devices, and used all the possibilities of the glove to control the system terminal. AssertRobot, developped in C++ with QT 3.3.3, allows to take some objects on the worktop, or to reproduce the mouvment of the glove on the robot .

Keywords

Robotic, SCORBOT, 3D sensor, P5 glove, automation.